

FIADB APIs
(Forest Inventory and Analysis Database - Application Programming Interfaces)
Pat Miles
May 15, 2018

Introduction

The Forest Inventory and Analysis Database was developed in 2001. From the beginning FIADB data has been publicly available on the Internet for downloading in CSV (comma-separated values) format. With the release of FIADB version 1.7.2.00 in May of 2018 eight FIADB Application Programming Interfaces (APIs) are now publicly available. These APIs were developed as Java RESTful web services with Jersey Framework.

From the practical standpoint there are really only four functions supported by the FIADB APIs. Each of these four functions can be called as a GET command or as a POST command, thus eight APIs. A GET API can be run directly from a browser address line whereas a POST API cannot. The primary reason for having a POST API is that there is virtually no limit to the amount of data that can be passed to the application whereas a GET API can only pass about 2000 characters to the application.

The four chapters and their APIs.

Chapter Number	API name	API Type	Description
1	fullreport	GET	Returns an EVALIDator report based on 22 input parameters.
1	fullreportPost	POST	Returns an EVALIDator report based on 22 input parameters.
2	evalgrp	GET	Returns a list of evaluation groups (in JSON format) meeting criteria.
2	evalgrpPost	POST	Returns a list of evaluation groups (in JSON format) meeting criteria.
3	statecdLonLatRad	GET	Returns a list of state FIPS codes for states that are at least partially within a circle defined by input parameters lon, lat, and rad.
3	statecdLonLatRadPost	POST	Returns a list of state FIPS codes for states that are at least partially within a circle defined by input parameters lon, lat, and rad.
4	refTable	GET	Returns information from any single publicly available FIADB table. Can return columns or aggregate function information. Output in HTML, XML or JSON formats.
4	refTablePost	POST	Returns information from any single publicly available FIADB table. Can return columns or aggregate function information. Output in HTML, XML or JSON formats.

Each chapter contains a table describing the input parameters to the APIs and example GET and POST commands. Appendix A provides an example of MS-Excel macro running multiple GET commands. Appendix B provides an example of a Java program running a POST command.

Additional APIs will be developed as the need arises. User feedback is appreciated.

Chapter 1. fullreport and fullreportPost

(Running EVALIDator reports via an API)

The EVALIDator web-application, developed in 2007, provides estimates and sampling errors of forest statistics (e.g., forest area, number of trees, tree biomass...) from data stored in the FIADB. There are two parts to the EVALIDator web-application: 1) a Graphical User Interface for collecting input parameters and, 2) a report generator that accepts the input parameters and returns a report. The “fullreport” and “fullreportPost” APIs run EVALIDator reports. These APIs bypass the EVALIDator Graphical User Interface by passing all of the input parameters to the report generator as part of the API call.

Because these API bypass the EVALIDator’s GUI there is an increased likelihood that the user will try to produce forest statistics that are not supported by the data. For example, the user could try to produce an estimate of the volume on forest land for the 1977 inventory of Minnesota. However, the 1977 Minnesota inventory predates the annual inventory implementation so tree measurements would not have been taken on reserved and unproductive forest land. The estimate returned would be based only on productive non-reserved forest land, thereby underreporting the desired number. It is suggested that a trial run of the EVALIDator GUI (<https://apps.fs.usda.gov/Evalidator/evalidator.jsp>) be used to verify an estimate is supported by the data. If the run is successful the resulting output will include the estimates and a “fullreport” API GET script that can be run to produce the same output.

The “fullreport” API requires 22 parameters (Table 1.1). An example of how this information can be used to generate an EVALIDator report is provided in Figure 1. This example contains the URL and the parameters necessary to generate an estimate of the “Area of timberland, in acres” by “Stand-size class”, and “Ownership group – Major”, within “50” miles of latitude “45” degrees North, and “-93” degrees West.

Table 1.1 EVALIDator API input parameters.

Parameter	Valid values
reptype	“State”, “Circle”
lat	0.0000 to 90.0000 (decimal degrees NAD83)
lon	-180.0000 to 180.0000 (decimal degrees NAD83)
radius	0.0 to 1000.0 (units are in miles)
snum	See values of attribute_descr variable in evaluator_pop_est_1_6_1 FIADB table
sdenom	If not performing a ratio estimate then enter “No denominator - just produce estimate.” For ratio estimates see values of attribute_descr variable in evaluator_pop_est_1_6_1 FIADB table.
wc	See values of eval_grp variable in pop_eval_grp FIADB table. When more than one evaluation group is selected the evaluation group numbers should be separated by a comma.

pselected	<p>See values of label_var variable in validator_variable_library table where page_list='Y'.</p> <p>If pages breakdowns are not desired enter "None"</p>
rselected	See values of label_var variable in validator_variable_library table where row_list='Y'.
cselected	See values of label_var variable in validator_variable_library table where col_list='Y'.
ptime	<p>"Accounting", "Previous", "Current", "Previous if available else current", "Current if available else previous"</p> <p>Note: Always use "Current" unless generating growth, removals or mortality estimates.</p>
rtime	<p>"Accounting", "Previous", "Current", "Previous if available else current", "Current if available else previous"</p> <p>Note: Always use "Current" unless generating growth, removals or mortality estimates.</p>
ctime	<p>"Accounting", "Previous", "Current", "Previous if available else current", "Current if available else previous"</p> <p>Note: Always use Current unless generating growth, removals or mortality estimates.</p>
wf	SQL clause filter used for non-ratio estimates.
wnum	SQL clause filter - only applied to numerator in a ratio estimate .
wnumdenom	SQL clause filter - applied to both numerator and denominator in a ratio estimate.
FIAorRPA	"FIADEF" or "RPADEF" The RPA definition uses a more restrictive definition of forest land
outputFormat	"HTML", "XML", "JSON"
estOnly	"Y", "N" If you enter "Y" the retrieval will run faster but you will not get sampling errors.
schemaName	"FS_FIADB." Note that there is a period at the end.
r1	This should only be used if you want to supply a list of plot control numbers and associated values to generate a report using those values for rows. Otherwise leave it blank. There must be a comma after each plot control number (CN) and after each value.
c1	This should only be used if you want to supply a list of plot control numbers and associated values to generate a report using those values for columns. Otherwise leave it blank. There must be a comma

after each plot control number (CN) and after each value.

```
https://apps.fs.usda.gov/Evaluator/rest/Evaluator/fullreport?reptype=Circle&lat=45&lon=-93&radius=50&snum=Area of timberland, in acres&sdenom=No denominator - just produce estimates&wc=272015,552015&pselected=None&rselected=Stand-size class&cselected=Ownership group - Major&ptime=Current&rtime=Current&ctime=Current&wf=&wnum=&wnumdenom=&FIAorRPA=FIADEF&outputFormat=HTML&estOnly=N&schemaName=FS_FIA_DB.
```

Figure 1.1. fullreport API GET call - Area of timberland, in acres” by “Stand-size class”, and “Ownership group – Major”, within 50 miles of latitude 45 degrees north, and longitude -93 degrees west for evaluation groups 272015 (Minnesota 2015) and 552015 (Wisconsin 2015). Output from this retrieval is in Appendix C.

Hostname= apps.fs.usda.gov

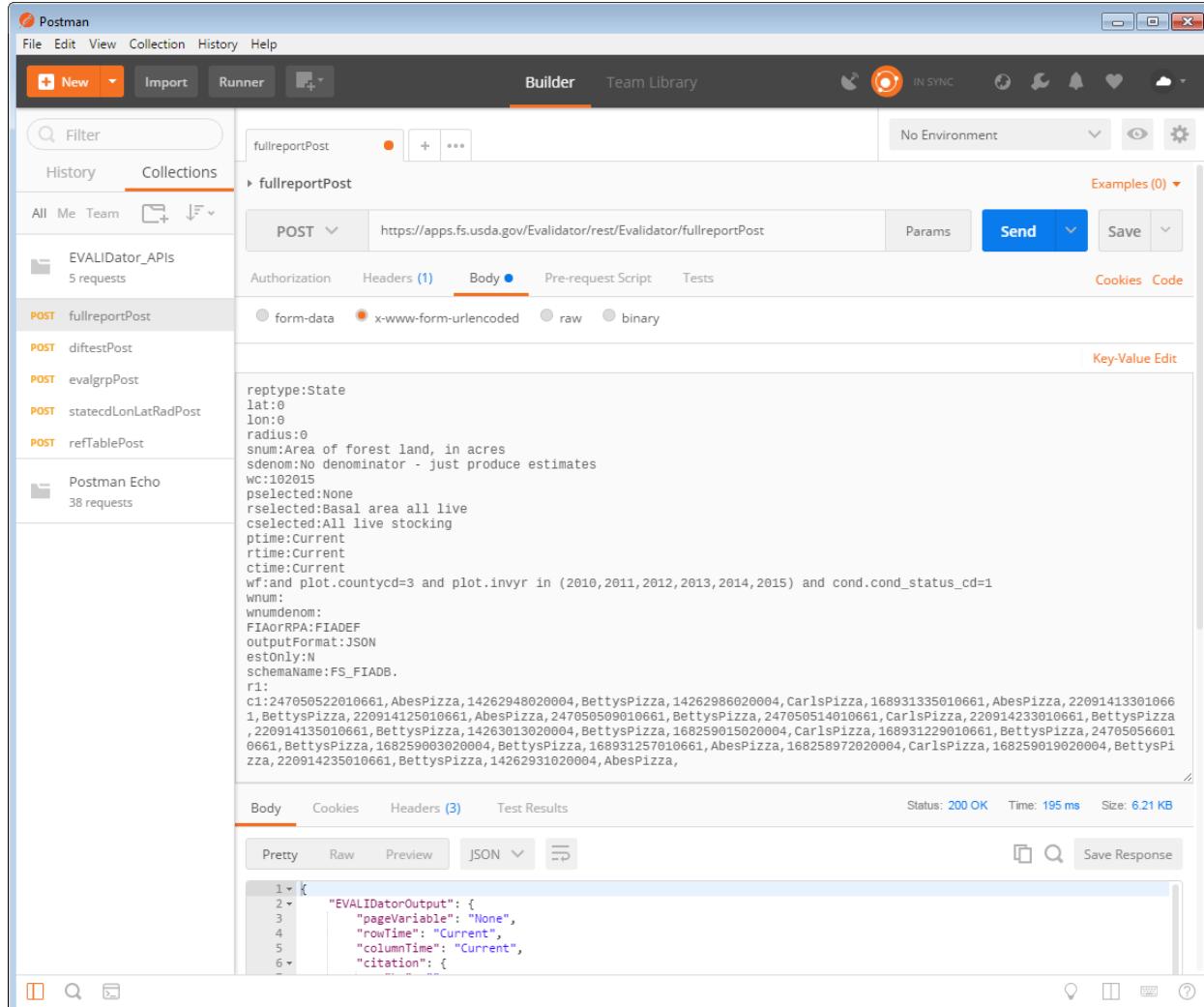
Instance=Evaluator

Protocol=rest

Service=Evaluator/fullreport

An example of how to run the fullreportPost POST API using the Postman(© Postdot Technologies Inc) program is provided in figure 1.2. POST APIs can handle tremendous amounts of data. This is useful when you want to report estimates using data that is not in the FIADB.

Figure 1.2. Example of a “fullreportPost” API POST call using POSTMAN desktop application.



Reprinted with permission © Postdot Technologies Inc. All rights reserved. Note: there is a comma after each plot control number (CN) and after each value in the C1 parameter.

In the example shown in figure 1.2 the county planner for New Castle County (countycd=3), Delaware, wanted to generate estimates of the area of forest land in New Castle county by PizzaDeliveryZone. Unfortunately PizzaDeliveryZone is not in the FIADB. However, latitude and longitude data is in the FIADB. So if the planner had a map of pizza delivery zones and could overlay FIA plot locations on that map, the planner could then assign PizzaDeliveryZone values to each FIA plot.

The county planner would assign all of the plot CNs and their PizzaDeliveryZone values to the “c1” variable (the 22nd and final parameter) in order to report forest area estimates by PizzaDeliveryZone. If the output format was HTML (Figure 1.3) then each PizzaDeliveryZone would have it’s own column.

EVALIDator Version 1.7.0.01 - View report

Numerator attribute number and description: 0002 Area of forest land, in acres
FIADEF as the forest land definition.

Statecd/EVALID(s):

Delaware 102015

Page variable=None (based on values from the Current inventory).

Row variable=Basal area all live (based on values from the Current inventory).

Column variable=User plot overlay values (based on values from the Current inventory).

Filtering clause(s): and plot.countycd=3 and plot.invyr in (2010,2011,2012,2013,2014,2015)
and cond.cond_status_cd=1

Estimate:

		User plot overlay values			
Basal area all live		Total	AbesPizza	BettysPizza	CarlsPizza
Total		59,139	17,881	30,507	10,751
0-40 sqft/ac		9,836	2,274	4,787	2,775
41-80 sqft/ac		905	905	-	-
81-120 sqft/ac		20,157	7,472	9,474	3,212
120+ sqft/ac		28,240	7,230	16,246	4,765

Figure 1.3. Format for the estimate output from the Postman retrieval in figure 2 if the outputFormat parameter had been set to “HTML” instead of “JSON”.

But suppose you want to produce dozens of reports. Copying and pasting dozens of URLs containing these 22 parameters would be awkward and time consuming. An MS-Excel macro was created to demonstrate how to run multiple reports using the fullreport GET API (see Appendix A). Java code is provided in Appendix B to illustrate how the fullreportPost API can be run from a Java application.

Chapter 2. evalgrp and evalgrpPost

The evalgrp and evalgrpPost APIs return output in JSON format. These APIs are used to identify which evaluation groups are in the database. The “evalgrp” API requires 3 input parameters (Table 2.1). An example of how this information can be used to report evaluation group information is provided in Figure 2.1. This example contains the URL and the parameters necessary to report the most recent evaluation groups for Minnesota and Wisconsin.

Table 2.1 evalgrp and evalgrpPost API input parameters.

Parameter	Valid values
schemaName	“FS_FIADB” Note that there is NOT a period at the end.
whereClause	Usually left blank but user can use any fields in the POP_EVAL_GRP table to limit the number of rows returned by this query.
mostRecent	“Y” or “N”. If “Y” then only the most recent inventories (based on information in the DATAMART_MOST_RECENT_INV table) will be returned.

[https://apps.fs.usda.gov/Validator/rest/Validator/evalgrp?schemaName=FS_FIADB&whereClause=statecd in \(27,55\)&mostRecent=Y](https://apps.fs.usda.gov/Validator/rest/Validator/evalgrp?schemaName=FS_FIADB&whereClause=statecd%20in%20(27,55)&mostRecent=Y)

Figure 2.1. evalgrp API GET call that will return the most recent inventories for Minnesota (statecd=27) and Wisconsin (statecd=55).

The output from the evalgrp GET call in figure 2.1 will be in JSON format and will look like this:

```
{"listOfEvalGroups": {"evalGrp": [ 272017, 552017 ]}}
```

An example of how to run the evalgrpPost POST API using the Postman(© Postdot Technologies Inc) program is provided in figure 2.2. This example contains the URL and the parameters necessary to report the most recent evaluation groups for Minnesota and Wisconsin.

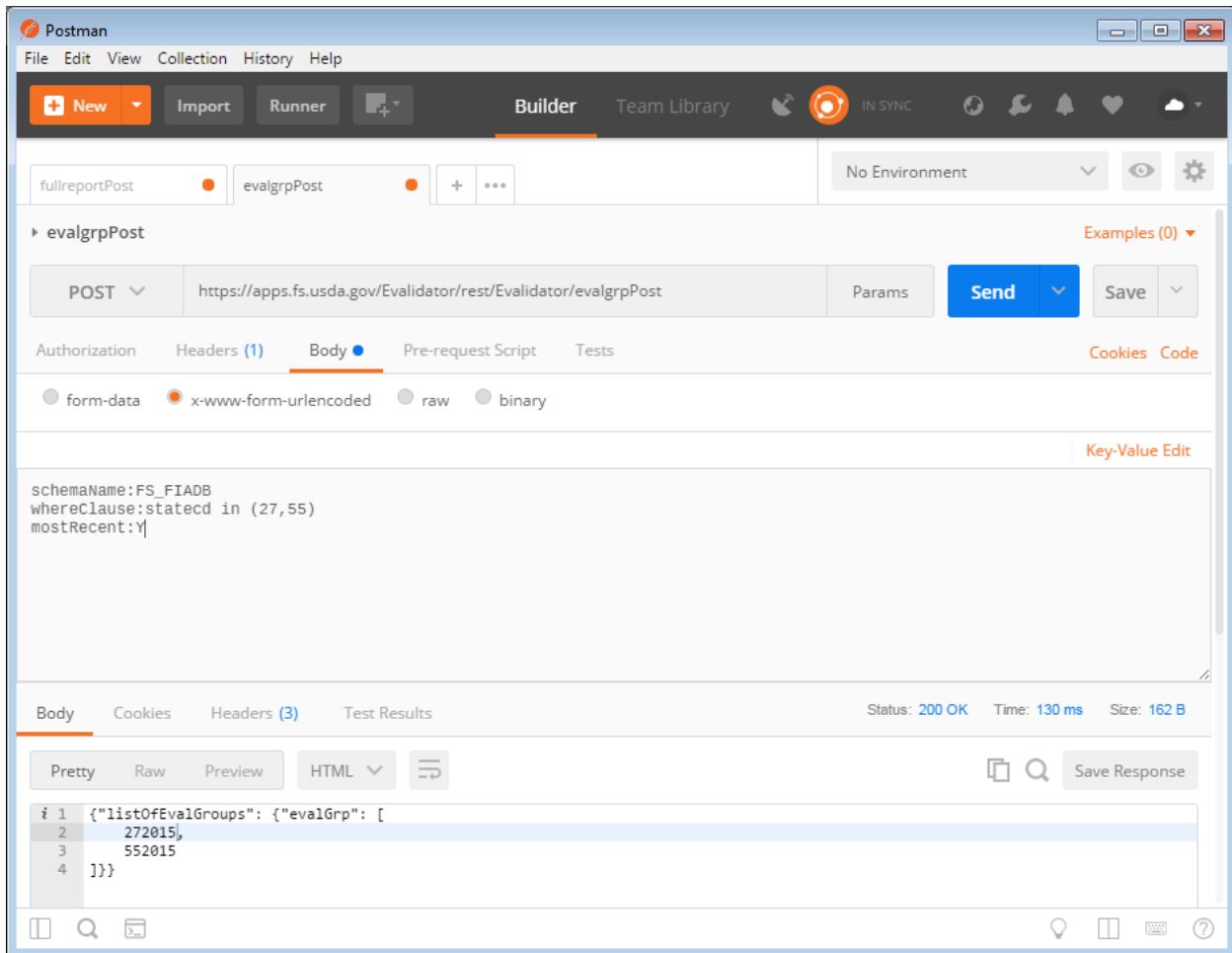


Figure 2.2 Example of a “evalgrpPost” API POST call using POSTMAN desktop application.
Reprinted with permission © Postdot Technologies Inc. All rights reserved.

Chapter 3. statecdLonLatRad and statecdLonLatRadPost

The statecdLonLatRad and statecdLonLatRadPost APIs return output in JSON format. These APIs are used to identify which states are within a specified radius from a specified point. The “statecdLonLatRad” API requires 4 input parameters (Table 3.1). An example of how this information can be used to report the state FIPS codes for all states whose boundaries fall within the specified radius (in miles) of a specified point (lat/lon coordinates NAD83) is provided in Figure 3.1. This example contains the URL and the parameters necessary to report those states falling within 100 miles of a point centered at -93 degrees longitude and 45 degrees latitude.

Table 3.1 evalgrp and evalgrpPost API input parameters.

Parameter	Valid values
schemaName	“FS_FIA_SPATIAL” Note that there is NOT a period at the end.
lon	Longitude NAD83 Note that all longitude values should be negative.
lat	Latitude NAD83
rad	Radius in miles

`https://apps.fs.usda.gov/Validator/rest/Validator/statecdLonLatRad?schemaName=FS_FIA_SPATIAL&lon=-93&lat=45.0&rad=100`

Figure 3.1. statecdLonLatRad API GET call that will return the FIPS state codes for those states that are within 100 miles of longitude -93 and latitude 45.

The output from the statecdLonLatRad GET call in figure 3.1 will be in JSON format and will look like this (55 is the FIPS state code for Wisconsin and 27 is the FIPS code for Minnesota):

```
{"listOfStatecds": {"statecd": [ 55, 27 ]}}
```

An example of how to run the statecdLonLatRadPost POST API using the Postman(© Postdot Technologies Inc) program is provided in figure 3.2. This example contains the URL and the parameters necessary to report FIP codes for those states whose boundaries fall within a circle centered at -93 degrees longitude and 45 degrees latitude with a radius of 100 miles.

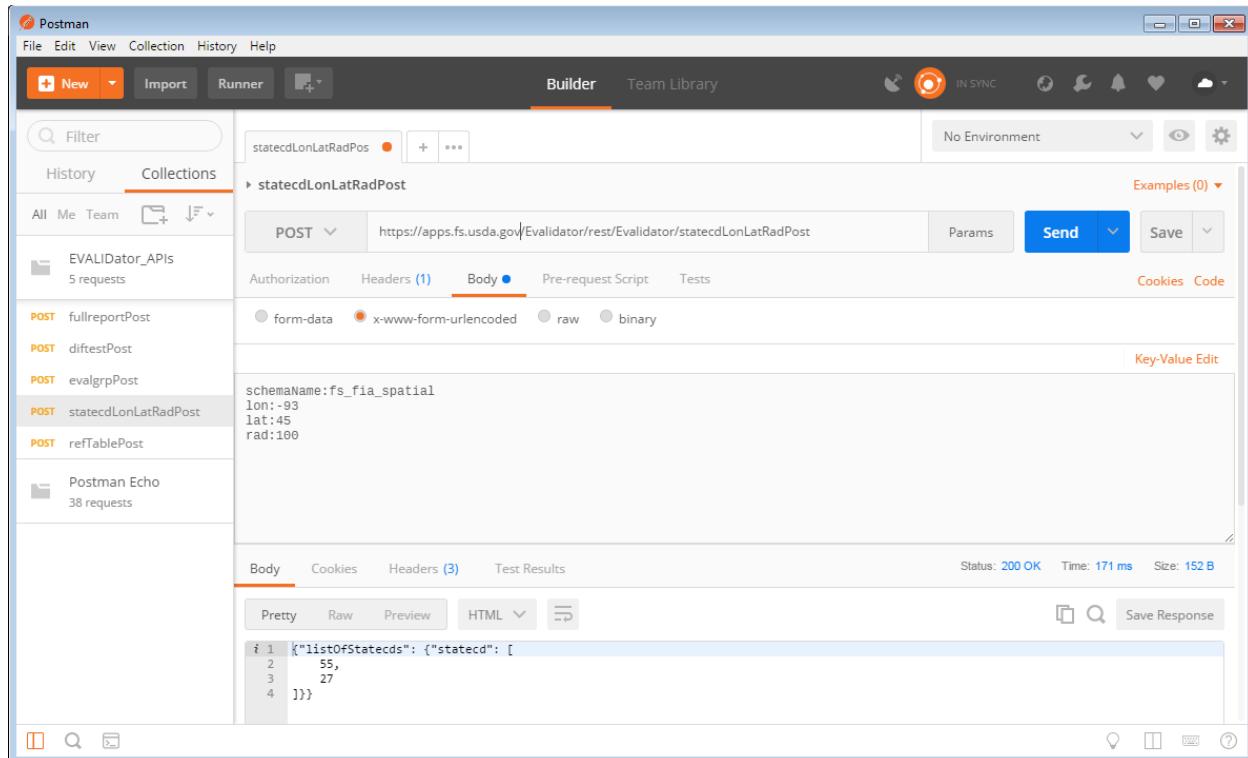


Figure 3.2 Example of a “statecdLonLatRadPost” API POST call using POSTMAN desktop application. Reprinted with permission © Postdot Technologies Inc. All rights reserved.

Chapter 4. refTable and refTablePost

The refTable and refTablePost APIs return output in HTML, XML or JSON formats. These APIs can be used to return information from any publicly available FIADB table. The “refTable” API requires 4 input parameters (Table 4.1). An example of how this information can be used to report the number of TREE table records for Minnesota is provided in Figure 4.1a. An example of how to report several field values for each record from the REF_SPECIES table where the common name contains the text string “ASPEN” is provided in Figure 4.1b.

Table 4.1 evalgrp and evalgrpPost API input parameters.

Parameter	Valid values
colList	This parameter can be a list of columns that are contained in the table selected (Note: the column names must be separated by commas. Use the “*” symbol to return all columns.) This parameter could be an aggregate function like “Count(*)” which would report the number of records that meet the criteria specified in the whereStr parameter.
tableName	Any publicly available FIADB table name
whereStr	Filter based on columns in the table selected
outputFormat	“HTML”, “XML”, or “JSON”

`https://apps.fs.usda.gov/Validator/rest/Validator/refTable?colList=count(*)&tableName=TREE&whereStr=STATECD=27&outputFormat=HTML`

Figure 4.1a. refTable API GET call that will return the number of records in the TREE table for Minnesota (statecd=27).

The output from the refTable GET call in figure 3.1 will be in HTML format and will look like this:

COUNT(*)
1398902

`https://apps.fs.usda.gov/Validator/rest/Validator/refTable?colList=common_name,%20genus,%20species&tableName=REF_SPECIES&whereStr=upper(common_name)%20like%20%27%ASPEN%27&outputFormat=HTML`

Figure 4.1b. refTable API GET call that will return data from the ref_species table where the uppercase value of common_name has the string “ASPEN” in it.

The output from the refTable GET call in figure 4.1b will be in HTML format and will look like this:

COMMON_NAME	GENUS	SPECIES
bigtooth aspen	Populus	grandidentata
quaking aspen	Populus	tremuloides

An example of how to run the refTable POST API using the Postman(© Postdot Technologies Inc) program is provided in figure 4.2. This example contains the URL and the parameters necessary to report the number of records in the REF_SPECIES table.

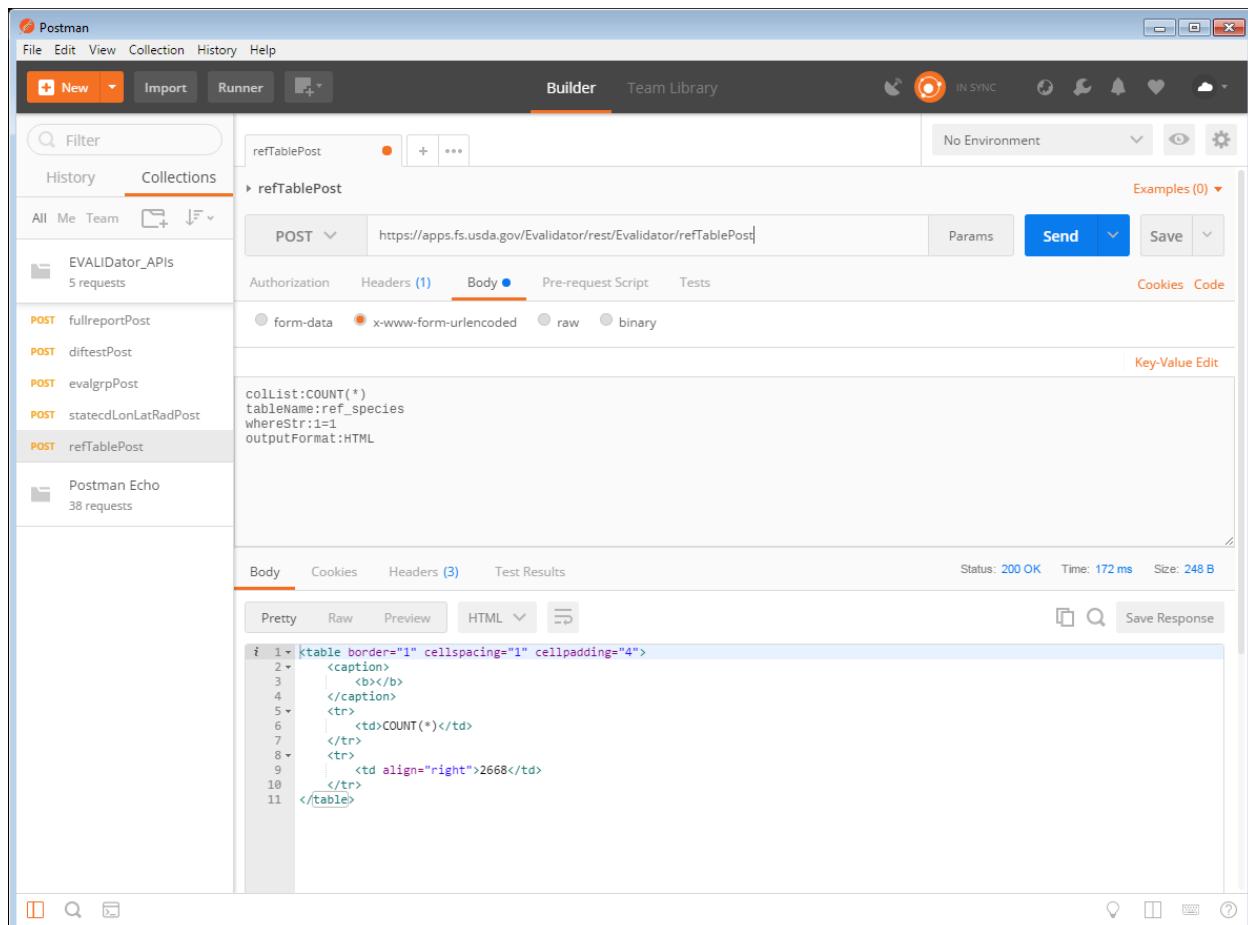


Figure 4.2 Example of a “refTablePost” API POST call using POSTMAN desktop application.
Reprinted with permission © Postdot Technologies Inc. All rights reserved.

APPENDIX A

Calling the fullreport GET API and the fullreportPost POST API from Microsoft Excel

MS-Excel workbooks are available for downloading from this webpage
https://apps.fs.usda.gov/fia/datamart/datamart_excel.html.

These Excel workbooks contain VBA macros that will send either GET (fullreport) or POST (fullreportPost) API commands to the database to run EVALIDator reports. Beginning with line 11 the user can enter EVALIDator input parameters. Each line will result in the generation of a separate API call and therefore a separate report (displayed on a separate worksheet in the workbook). In the example in Figure A1 we know there are at least 10 reports as lines 11 through 20 have input parameters.

Figure A1. MS-Excel Workbook push “Query” button to initiate EVALIDator runs.

Table A1. MS-Excel VBA source that calls fullreportPost API.

```
Sub fullreportPost()
    Dim tablename As String
    Dim reptype As String
    Dim lat As Double
    Dim lon As Double
    Dim radius As Double
    Dim snum As String
```

```

Dim sdenom As String
Dim wc As String
Dim pselected As String
Dim rselected As String
Dim cselected As String
Dim ptime As String
Dim rtime As String
Dim ctime As String
Dim wf As String
Dim wnum As String
Dim wnumdenom As String
Dim FIAorRPA As String
Dim outputFormat As String
Dim estOnly As String
Dim schemaName As String
Dim mill_id As String
Dim sURL As String
Dim sResult As String
Dim urlStr As String
Dim startPage As String
Dim endPage As String
Dim workbook_name As String
Dim worksheet_name(1000) As String

For i = 11 To 1011 'maximum number of reports is set to 1000 so max=1011
    icnt = i - 10
    tablenm = ThisWorkbook.Sheets("INPUTS").Range("a" + LTrim(Str(i)))
    reptype = ThisWorkbook.Sheets("INPUTS").Range("b" + LTrim(Str(i)))
    lat = ThisWorkbook.Sheets("INPUTS").Range("c" + LTrim(Str(i)))
    lon = ThisWorkbook.Sheets("INPUTS").Range("d" + LTrim(Str(i)))
    radius = ThisWorkbook.Sheets("INPUTS").Range("e" + LTrim(Str(i)))
    snum = ThisWorkbook.Sheets("INPUTS").Range("f" + LTrim(Str(i)))
    sdenom = ThisWorkbook.Sheets("INPUTS").Range("g" + LTrim(Str(i)))
    wc = ThisWorkbook.Sheets("INPUTS").Range("h" + LTrim(Str(i)))
    pselected = ThisWorkbook.Sheets("INPUTS").Range("i" + LTrim(Str(i)))
    rselected = ThisWorkbook.Sheets("INPUTS").Range("j" + LTrim(Str(i)))
    cselected = ThisWorkbook.Sheets("INPUTS").Range("k" + LTrim(Str(i)))
    ptime = ThisWorkbook.Sheets("INPUTS").Range("l" + LTrim(Str(i)))
    rtime = ThisWorkbook.Sheets("INPUTS").Range("m" + LTrim(Str(i)))
    ctime = ThisWorkbook.Sheets("INPUTS").Range("n" + LTrim(Str(i)))
    wf = ThisWorkbook.Sheets("INPUTS").Range("o" + LTrim(Str(i)))
    wnum = ThisWorkbook.Sheets("INPUTS").Range("p" + LTrim(Str(i)))
    wnumdenom = ThisWorkbook.Sheets("INPUTS").Range("q" + LTrim(Str(i)))
    FIAorRPA = ThisWorkbook.Sheets("INPUTS").Range("r" + LTrim(Str(i)))
    outputFormat = ThisWorkbook.Sheets("INPUTS").Range("s" + LTrim(Str(i)))
    estOnly = ThisWorkbook.Sheets("INPUTS").Range("t" + LTrim(Str(i)))
    schemaName = ThisWorkbook.Sheets("INPUTS").Range("u" + LTrim(Str(i)))
If tablenm <> "" Then
    urlStr = "reptype=" + reptype
    urlStr = urlStr + "&lat=" + Str(lat)
    urlStr = urlStr + "&lon=" + Str(lon)
    urlStr = urlStr + "&radius=" + Str(radius)
    urlStr = urlStr + "&snum=" + snum
    urlStr = urlStr + "&sdenom=" + sdenom
    urlStr = urlStr + "&wc=" + wc
    urlStr = urlStr + "&pselected=" + pselected
    urlStr = urlStr + "&rselected=" + rselected
    urlStr = urlStr + "&cselected=" + cselected
    urlStr = urlStr + "&ptime=" + ptime
    urlStr = urlStr + "&rtime=" + rtime
    urlStr = urlStr + "&ctime=" + ctime
    urlStr = urlStr + "&wf=" + wf
    urlStr = urlStr + "&wnum=" + wnum
    urlStr = urlStr + "&wnumdenom=" + wnumdenom
    urlStr = urlStr + "&FIAorRPA=" + FIAorRPA
    urlStr = urlStr + "&outputFormat=" + outputFormat
    urlStr = urlStr + "&estOnly=" + estOnly
    urlStr = urlStr + "&schemaName=" + schemaName
    sResult = GetHTTPResult(urlStr)

```

```

startPage = "<!DOCTYPE html PUBLIC ""-//W3C//DTD XHTML 1.0 Transitional//EN"" ""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"">""
startPage = startPage + "<html lang=""en"" xmlns=""http://www.w3.org/1999/xhtml""><head></head><body>"
endPage = "</body></html>"

sResult = startPage + sResult + endPage

    worksheet_name(icnt) = tablenm
    Call OpenTextFile(worksheet_name(icnt), sResult)
    Else
        icnt = icnt - 1
        Exit For
    End If
    Next i

Call CombineSheets(Application.ActiveWorkbook.Path, workbook_name, worksheet_name, icnt)

'cleanup - delete *.html and *.xls files
Dim html_file As String
Dim xls_file As String
For i = 1 To icnt
    'Check that file exists
    html_file = Application.ActiveWorkbook.Path + "\" + worksheet_name(i) + ".html"
    xls_file = Application.ActiveWorkbook.Path + "\" + worksheet_name(i) + ".xls"
    If Len(Dir$(worksheet_name(i) + ".html")) > 0 Then
        SetAttr html_file, vbNormal
        Kill html_file
    End If
    If Len(Dir$(worksheet_name(i) + ".xls")) > 0 Then
        SetAttr xls_file, vbNormal
        Kill xls_file
    End If
    Next i

    ActiveWorkbook.Save
    Worksheets("TOC").Activate

End Sub
Function GetHTTPResult(sURL As String) As String
Set objHttp = CreateObject("MSXML2.ServerXMLHTTP")
URL = "https://apps.fs.usda.gov/Evalidator/rest/Evalidator/fullreportPost"
objHttp.Open "POST", URL, False
objHttp.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"
Debug.Print sURL
objHttp.Send sURL
GetHTTPResult = objHttp.responseText
End Function

Sub OpenTextFile(mill_id As String, sResult As String)
Dim File_Path As String, Folder_Path As String
Dim File_Path_Workbook As String
File_Path = Application.ActiveWorkbook.Path + "\" + mill_id + ".html"
File_Path_Workbook = Application.ActiveWorkbook.Path + "\" + mill_id + ".xls"

    'Check that file exists
    If Len(Dir$(File_Path)) > 0 Then
        SetAttr File_Path, vbNormal
        Kill File_Path
    End If
    If Len(Dir$(File_Path_Workbook)) > 0 Then
        SetAttr File_Path_Workbook, vbNormal
        Kill File_Path_Workbook
    End If

    Open File_Path For Output As #1
    Write #1, sResult
    Close #1
End Sub

```

```

Dim wb As Excel.Workbook
Set wb = Workbooks.Open(File_Path)
    wb.SaveAs File_Path_Workbook, FileFormat:=xlWorkbookNormal
    wb.Close
End Sub

Sub CombineSheets(sPath, workbook_name, worksheet_name, icnt)
    Dim sFName As String
    Dim wBk As Workbook
    Dim wSht As Variant
    Dim i As Integer
    Dim II As Integer

    Dim Folder_Path As String
    Folder_Path = Application.ActiveWorkbook.Path

    Application.EnableEvents = False
    Application.ScreenUpdating = False

    Application.DisplayAlerts = False
    For II = ThisWorkbook.Sheets.Count To 3 Step -1
        ThisWorkbook.Sheets(II).Delete
    Next II
    ThisWorkbook.Save
    ChDir Folder_Path
    For i = 1 To icnt
        sFname = worksheet_name(i) & ".xls"
        Set wBk = Workbooks.Open(sFname)
        Windows(sFname).Activate
        Sheets(wBk.Worksheets(1).Name).Copy After:=ThisWorkbook.Sheets(i + 1)
        wBk.Close False
    Next i
    ActiveWorkbook.Save
    Application.EnableEvents = True
    Application.ScreenUpdating = True

    'format the Table Of Contents page which is blank at this point
    Sheets("TOC").Select
    Sheets("TOC").Cells.ClearContents

    'cells(1,2) means cells(row=1, col=2)
    Cells(1, 1) = "USDA Forest Service, Forest Inventory and Analysis Program"
    Cells(2, 1) = "Run date:"
    Cells(2, 2) = Now
    Cells(4, 1) = "Table"
    Cells(4, 2) = "Numerator"
    Cells(4, 3) = "Denominator"
    Cells(4, 4) = "Page"
    Cells(4, 5) = "Row"
    Cells(4, 6) = "Column"
    Columns(1).Insert
    For i = 3 To Sheets.Count
        Cells(i + 2, 2) = Mid(Sheets(i).Cells(1, 1), 45, Len(Sheets(i).Cells(1, 1))) 'numerator type
        If (Left(Sheets(i).Cells(2, 1), 46) = "Denominator attribute number and description: ") Then
            Cells(i + 2, 3) = Mid(Sheets(i).Cells(2, 1), 47, Len(Sheets(i).Cells(2, 1))) 'denominator type
        End If
        For j = 1 To 15
            If (Left(Sheets(i).Cells(j, 1), 14) = "Page variable=") Then
                Cells(i + 2, 4) = RTrim(Mid(Sheets(i).Cells(j, 1), 15, 100))
            End If
            If (Left(Sheets(i).Cells(j, 1), 13) = "Row variable=") Then
                Cells(i + 2, 5) = RTrim(Mid(Sheets(i).Cells(j, 1), 14, 100))
            End If
            If (Left(Sheets(i).Cells(j, 1), 16) = "Column variable=") Then
                Cells(i + 2, 6) = RTrim(Mid(Sheets(i).Cells(j, 1), 17, 100))
            End If
        Next j
        Sheets("TOC").Cells(i + 2, 1).Select
    
```

```

Sheets(i).Hyperlinks.Add Anchor:=Selection, Address:="", _
    SubAddress:=Sheets(i).Name & "!A1", _
    TextToDisplay:=Sheets(i).Name
Range(Cells(i + 2, 1), Cells(i, 6)).WrapText = True
Range(Cells(i + 2, 1), Cells(i, 6)).ColumnWidth = 30

Next i
Call formatWS(Sheets.Count)

End Sub
Sub formatWS(tot_sheets)
'format each worksheet other than the TOC
'cells(1,2) means cells(row=1, col=2)
Dim worksheetExists As Boolean
Dim colChar(50) As String
Dim wsName(500) As String
Dim retVal As Boolean
Dim fontColor, backgroundColor As Integer
Dim valA(65536), valB(65536) As String
Dim startBox, endBox As Long
Dim est, filt, geog, page, row, col, title As String
Dim pathStr As String
Dim sheetName As String
Dim i As Long
Dim lineStr As String
Dim rrow As String
Dim fName(51) As String
Dim rangeStr, rangeString, cellStr, iniStr As String
Dim gsStr As String
Dim firstLineWithGridline As Integer
Dim lastLineWithGridline As Integer
Dim column As Integer
Dim numcols As Integer
numcols = 2
For k = 3 To tot_sheets
    'Format each worksheet
    Dim II As Integer
    Sheets(k).Select

    For i = 1 To 64000

        'for all of the lines in the spreadsheet find those that should have gridlines
        If (Left(Sheets(k).Cells(i, 1), 16) = "Filtering clause") Then
            firstLineWithGridline = i + 2
        ElseIf (Sheets(k).Cells(i, 1) = "Population Estimate Description" Or _
            Sheets(k).Cells(i, 1) = "Numerator Estimate Description") Then
            lastLineWithGridline = i - 2
        End If

        'put green, blue or red tabs on est, se, numplot table headers
        If (Sheets(k).Cells(i, 1) = "Estimate:" Or _
            Sheets(k).Cells(i, 1) = "Ratio estimate:") Then
            Range(Cells(i - 1, 1), Cells(i, 1)).Interior.Color = RGB(0, 255, 0)

        'determine number of columns the gridlines should surround
        For column = 1 To 51
            If (Sheets(k).Cells(i + 3, column) = "") Then
                numcols = column - 1
                Exit For
            End If
        Next column

        ElseIf (Sheets(k).Cells(i, 1) = "Numerator estimate:" Or _
            Sheets(k).Cells(i, 1) = "Denominator estimate:") Then
            Range(Cells(i - 1, 1), Cells(i, 1)).Interior.Color = RGB(0, 255, 0)
        ElseIf Sheets(k).Cells(i, 1) = "Sampling error percent (Confidence level 68%):" Then
            Range(Cells(i - 1, 1), Cells(i, 1)).Interior.Color = RGB(0, 0, 255)
            Range(Cells(i - 1, 1), Cells(i, 1)).Font.Color = RGB(255, 255, 255)
        ElseIf Sheets(k).Cells(i, 1) = "Number of non-zero plots numerator:" Then
            Range(Cells(i - 1, 1), Cells(i, 1)).Interior.Color = RGB(255, 0, 0)
    End If
End Sub

```

```

        Range(Cells(i - 1, 1), Cells(i, 1)).Font.Color = RGB(255, 255, 255)
    ElseIf (Sheets(k).Cells(i, 1) = "Number of non-zero plots in estimate:" Or _
        Sheets(k).Cells(i, 1) = "Number of non-zero plots denominator:") Then
        Range(Cells(i - 1, 1), Cells(i, 1)).Interior.Color = RGB(255, 0, 0)
        Range(Cells(i - 1, 1), Cells(i, 1)).Font.Color = RGB(255, 255, 255)
    ElseIf (Sheets(k).Cells(i, 1) = "Web citation:") Then
        Exit For
    End If
    Next i

'set gridlines
    Range(Cells(firstLineWithGridline, 1), Cells(lastLineWithGridline, numcols)).Select
    With Selection
        With .Borders(xlInsideVertical)
            .LineStyle = xlContinuous
            .Weight = xlThin
            .Color = RGB(0, 0, 0)
        End With

        With .Borders(xlInsideHorizontal)
            .LineStyle = xlContinuous
            .Weight = xlThin
            .Color = RGB(0, 0, 0)
        End With
        Range(Cells(firstLineWithGridline, 1), Cells(lastLineWithGridline, numcols)).BorderAround ColorIndex:=1, Weight:=xlThin
    End With
    Next k
    Exit Sub
End Sub

```

APPENDIX B Calling the fullreportPost POST API from a Java program.

Table B1. Java source code

```

package PostPkg;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.HttpClientBuilder;
import org.apache.http.message.BasicNameValuePair;
public class HttpURLConnectionExample {
    public static void main(String[] args) throws ClientProtocolException, IOException {
        HttpClient client = HttpClientBuilder.create().build();
        HttpPost post = new HttpPost(

```

```
    " https://apps.fs.usda.gov/Validator/rest/Validator/fullreportPost?");  
String r1=""; //r1 contains a list of plot CNs and ecosubsection codes for 4 plots  
r1=r1+"9187278010661, 232Hd,";  
r1=r1+"9187313010661, 232Ac,";  
r1=r1+"9187346010661, 232Hd,";  
r1=r1+"249446195489998, 232Hc,";  
List<BasicNameValuePair> nameValuePairs = new ArrayList<BasicNameValuePair>(1);  
nameValuePairs.add(new BasicNameValuePair("reptype", "State"));  
nameValuePairs.add(new BasicNameValuePair("lat", "0"));  
nameValuePairs.add(new BasicNameValuePair("lon", "0"));  
nameValuePairs.add(new BasicNameValuePair("radius", "0"));  
nameValuePairs.add(new BasicNameValuePair("snum", "Area of forest land, in acres"));  
nameValuePairs.add(new BasicNameValuePair("sdenom", "No denominator - just produce estimates"));  
nameValuePairs.add(new BasicNameValuePair("wc", "102015"));  
nameValuePairs.add(new BasicNameValuePair("pselected", "None"));  
nameValuePairs.add(new BasicNameValuePair("rselected", "Basal area all live"));  
nameValuePairs.add(new BasicNameValuePair("cselected", "All live stocking"));  
nameValuePairs.add(new BasicNameValuePair("ptime", "Current"));  
nameValuePairs.add(new BasicNameValuePair("rtime", "Current"));  
nameValuePairs.add(new BasicNameValuePair("ctime", "Current"));  
nameValuePairs.add(new BasicNameValuePair("wf", ""));  
nameValuePairs.add(new BasicNameValuePair("wnum", ""));  
nameValuePairs.add(new BasicNameValuePair("wnumdenom", ""));  
nameValuePairs.add(new BasicNameValuePair("FIAorRPA", "FIADEF"));  
nameValuePairs.add(new BasicNameValuePair("outputFormat", "JSON"));  
nameValuePairs.add(new BasicNameValuePair("estOnly", "N"));  
nameValuePairs.add(new BasicNameValuePair("schemaName", "FS_FIADB."));  
nameValuePairs.add(new BasicNameValuePair("r1", r1));  
nameValuePairs.add(new BasicNameValuePair("c1", ""));  
post.setEntity(new UrlEncodedFormEntity(nameValuePairs));  
HttpResponse response = client.execute(post);  
BufferedReader rd = new BufferedReader(new  
InputStreamReader(response.getEntity().getContent()));  
String line = "";  
while ((line = rd.readLine()) != null) {  
    System.out.println(line);  
}  
}  
}
```

APPENDIX C - Output from fullreport GET API script example in Figure 1.1.

This appendix contains the results from executing the following GET API script from a browser address line.

```
https://apps.fs.usda.gov/Validator/rest/Validator/fullreport?reptype=Circle&lat=45&lon=-93&radius=50&snum=Area of timberland, in acres&sdenom=No denominator - just produce estimates&wc=272015,552015&pselected=None&rselected=Stand-size class&cselected=Ownership group - Major&ptime=Current&rtime=Current&ctime=Current&wf=&wnum=&wnumdenom=&FIAo rRPA=FIADEF&outputFormat=HTML&estOnly=N&schemaName=FS_FIADB.
```

Numerator attribute number and description: 0003 Area of timberland, in acres FIADEF as the forest land definition.

Statecd/EVALID(s):

Minnesota 272015

Wisconsin 552015

Page variable=None (based on values from the Current inventory).

Row variable=Stand-size class (based on values from the Current inventory).

Column variable=Ownership group - Major (based on values from the Current inventory).

Circle retrieval centered at 45 degrees north and -93 degrees west with a radius of 50 miles.

Filtering clause(s):

Estimate:			
	Ownership group - Major		
Stand-size class	Total	Public	Private
Total	1,039,687	78,555	961,132
Large diameter	551,171	31,850	519,321
Medium diameter	301,534	19,998	281,536
Small diameter	174,740	26,707	148,033

Nonstocked	12,242	-	12,242
-------------------	--------	---	--------

Sampling error percent (Confidence level 68%):

Note: for 95% confidence level multiply SE pct by 1.96

Stand-size class	Ownership group - Major		
	Total	Public	Private
Total	4.80	17.15	5.01
Large diameter	6.59	25.85	6.82
Medium diameter	8.88	31.77	9.24
Small diameter	12.07	29.44	13.18
Nonstocked	39.91	-	39.91

Number of non-zero plots in estimate:

Note: total number of plots in selected evaluations=29390

Stand-size class	Ownership group - Major		
	Total	Public	Private
Total	444	38	409
Large diameter	246	18	228

Medium diameter	151	11	140
Small diameter	80	14	67
Nonstocked	8	-	8

Population Estimate Description

Forest land: Land at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. Also included are pinyon-juniper and chaparral areas in the West and afforested areas. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outermost edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide.

Timberland: Forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are included.)

Page classification variable description

No pages selected.

Row classification variable description

Stand-size class: A classification of forest land based on the size class of live trees in the area.

The classes are as follows:

Nonstocked: Forest land stocked with less than 10 percent of full stocking with live trees.

Examples are recently cutover areas or recently reverted agricultural fields.

Seedling-sapling: Forest land stocked with at least 10 percent of full stocking with live trees with half or more of such stocking in seedlings or saplings or both.

Poletimber: Forest land stocked with at least 10 percent of full stocking with live trees with half or more of such stocking in poletimber or sawtimber trees or both, and in which the stocking of poletimber exceeds that of sawtimber.

Sawtimber: Forest land stocked with at least 10 percent of full stocking with live trees with half or more of such stocking in poletimber or sawtimber trees or both, and in which the stocking of sawtimber is at least equal to that of poletimber.

Column classification variable description

Owner group major - ownership codes are collapsed into Public and Private categories.

Sampling design/estimation method: post-stratification, as described in:

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p.

Web citation:

USDA Forest Service, Forest Inventory and Analysis Program, Thu Feb 15 15:53:43 GMT 2018. Forest Inventory EVALIDator web-application Version 1.7.0.01. St. Paul, MN: U.S. Department of Agriculture, Forest Service, Northern Research Station. [Available only on internet:
<http://fsxopsx0523.wrk.fs.usda.gov:30000/Validator/validator.jsp>]

SQL Script

This SQL script can be used to derive estimates from FIA Data Base (FIADB)

```
select coalesce(pagestr,`0000 Page total`),coalesce(rowstr,`0000 Row  
total`),coalesce(colstr,`0000 Column total`),sum(estimated_value) from (SELECT case 1 when  
1 then `0001 None` end as pagestr, decode(nvl(cond.stdszcd,-1), 1,`0001 Large diameter`,  
2,`0002 Medium diameter`, 3,`0003 Small diameter`, 5,`0004 Nonstocked`, -1,`0005 Not  
collected`, `0006 Other`) as rowstr, decode(cond.OWNGRPCD, 10,`0001 Public`, 20,`0001  
Public`, 30,`0001 Public`, 40,`0002 Private`, `0003 Unknown`) as colstr,  
SUM((COND.CONDPROP_UNADJ * CASE COND.PROP_BASIS WHEN 'MACR' THEN  
POP_STRATUM.ADJ_FACTOR_MACR ELSE POP_STRATUM.ADJ_FACTOR_SUBP  
END)*POP_STRATUM.EXPNS) AS ESTIMATED_VALUE FROM  
FS_FIADB.POP_STRATUM POP_STRATUM join  
FS_FIADB.POP_PLOT_STRATUM_ASSGN on  
(POP_PLOT_STRATUM_ASSGN.STRATUM_CN = POP_STRATUM.CN) join  
FS_FIADB.PLOT on (POP_PLOT_STRATUM_ASSGN.PLT_CN = PLOT.CN) join  
FS_FIADB.PLOTGEOM on (PLOT.CN = PLOTGEOM.CN) JOIN FS_FIADB.COND ON  
(COND.PLT_CN = PLOT.CN) WHERE COND.RESERVCD = 0 AND COND.SITECLCD IN  
(1, 2, 3, 4, 5, 6) AND COND.COND_STATUS_CD = 1 AND COND.CONDPROP_UNADJ IS  
NOT NULL AND ((pop_stratum.rscd=23 and pop_stratum.evalid=271501) or  
(pop_stratum.rscd=23 and pop_stratum.evalid=551501)) and plot.cn in (select p.cn from  
FS_FIADB.plot p, FS_FIADB.pop_plot_stratum_assgn q where p.cn=q.plt_cn and
```

```
q.rscd=pop_stratum.rscd and q.evalid=pop_stratum.evalid and ((pop_stratum.rscd=23 and pop_stratum.evalid=271501) or (pop_stratum.rscd=23 and pop_stratum.evalid=551501)) and p.lat>44.242424242424 and p.lat<45.757575757576 and p.lon<-90.61904761904762 and p.lon>-95.38095238095238 and SDO_GEOGRAPHY_DISTANCE(SDO_GEOGRAPHY(2001, 8265, SDO_POINT_TYPE(-93.0, 45.0, NULL), NULL, NULL), SDO_GEOGRAPHY(2001, 8265, SDO_POINT_TYPE(p.lon, p.lat, NULL), NULL, NULL), 0.0001, 'unit=mile') <= 50) GROUP BY case 1 when 1 then '^0001 None` end, decode(nvl(cond.stdszcd, -1), 1, '^0001 Large diameter', 2, '^0002 Medium diameter', 3, '^0003 Small diameter', 5, '^0004 Nonstocked', -1, '^0005 Not collected', '^0006 Other'), decode(cond.OWNINGRP_CD, 10, '^0001 Public', 20, '^0001 Public', 30, '^0001 Public', 40, '^0002 Private', '^0003 Unknown') ) tmpzzz group by cube(pagestr, rowstr, colstr) order by pagestr, rowstr, colstr
```

RESTful Web Service Call

This RESTful Web Service GET call can be used to generate estimates directly from browser address line

```
http://fsxopsx0523.wrk.fs.usda.gov:30000/Validator/rest/Validator/fullreport?reptype=Circle&lat=45&lon=-93&radius=50&snum=Area of timberland, in acres&sdenom=No denominator - just produce estimates&wc=272015,552015&pselected=None&rselected=Stand-size class&cselected=Ownership group - Major&ptime=Current&rtime=Current&ctime=Current&wf=&wnum=&wnumdenom=&FIAorRPA=FIADEF&outputFormat=HTML&estOnly=N&schemaName=FS_FIADB.
```

Start time Thu Feb 15 15:52:41 GMT 2018

End time Thu Feb 15 15:53:43 GMT 2018

APPENDIX D – Output from difftest GET API script example in Figure 5.1.

Difference Tester Version 1.7.0.01 - View report

AREA: Area of sampled land and water, in acres	
Description	Value
Created date	2018-02-16 11:31:34.0
Schema	FS_FIADB
Estimation type	0079_Y AREA: Area of sampled land and water, in acres
Across Eval Groups	102015
CONSTRAINT_X_TIME2	and c.cond_status_cd = 1
CONSTRAINT_Y_TIME1	and c1.cond_status_cd = 1
ESTIMATE_X	360279.8108174375
ESTIMATE_Y	365644.0880670985
DIFFERENCE_X_Y	-5364.277
DIFFERENCE_TEST	P-value is less than critical value of 1 standard error associated with 68% confidence level
STD_ERROR_DIFFERENCE	4794.397522108487
Z_SCORE	-1.1188636268193486381595895760220738585
P_VALUE_TWO_TAILED_TEST	.262714
NON_ZERO_PLOTS_X	122
NON_ZERO_PLOTS_Y	124

VARIANCE_X	2.0613187880369145E8
VARIANCE_Y	2.1159380023148662E8
COVARIANCE_XY_FINAL	1.973697157172353E8
COV_CORRECTION	1.0
TOTAL_PLOTS	374
TOTAL_POPULATION_ACRES	1296832.3

Web citation:

Pugh, S.A.; Westfall, J.A. Fri Feb 16 11:31:34 CST 2018. Difference Tester web-application Version 1.7.0.01. St. Paul, MN: U.S. Department of Agriculture, Forest Service, Northern Research Station. [Available only on internet:
http://localhost:8080/Validator/rest/Validator/diftest?estType=0079_Y&evalGrp=102015&constraintxT2=and%20c.cond_status_cd%20=%201&constraintyT1=and%20c1.cond_status_cd%20=%201&schemaName=FS_FIADB.&outputFormat=HTML/Validator/diftest.jsp]

SQL Script

This SQL statement can be used to derive estimates:

```
select sysdate created_date, 'FS_FIADB' schema, '0079_Y AREA: Area of sampled land and water, in acres' Comparison, '102015'
Across_Eval_Groups, 'and c.cond_status_cd = 1' constraint_x_time2, 'and c1.cond_status_cd = 1' constraint_y_time1, estimate_x, estimate_y,
difference_x_y, case when std_error_difference = 0 then 'Standard error of difference is 0' when round((abs(difference_x_y - 0) - (2.58 *
std_error_difference)), 3) > 0 then 'P-value is less than critical value of 3 standard errors associated with 99% confidence level' when
round((abs(difference_x_y - 0) - (2.58 * std_error_difference)), 3) > 0 then 'P-value is less than critical value of 2 standard errors associated
with 95% confidence level' when round((abs(difference_x_y - 0) - (2.58 * std_error_difference)), 3) > 0 then 'P-value is less than critical value of 1 standard error associated
with 68% confidence level' else 'P-value is greater than critical value of 1 standard error' end Difference_Test,
std_error_difference, case when std_error_difference <> 0 then to_char(((difference_x_y - 0) / std_error_difference)) else 'Standard error of
difference is 0' end z_score, case when std_error_difference = 0 then 'Standard error of difference is 0' when (abs(difference_x_y - 0) /
std_error_difference) > 3.99 THEN '<.000066' else to_char((round((2.0 * DECODE(round(abs((difference_x_y - 0) / std_error_difference), 2),
3.99, 0.000033, 3.98, 0.000034, 3.97, 0.000036, 3.96, 0.000037, 3.95, 0.000039, 3.94, 0.000041, 3.93, 0.000042, 3.92, 0.000044, 3.91,
0.000046, 3.90, 0.000048, 3.89, 0.000050, 3.88, 0.000052, 3.87, 0.000054, 3.86, 0.000057, 3.85, 0.000059, 3.84, 0.000062, 3.83, 0.000064,
3.82, 0.000067, 3.81, 0.000069, 3.80, 0.000072, 3.79, 0.000075, 3.78, 0.000078, 3.77, 0.000082, 3.76, 0.000085, 3.75, 0.000088, 3.74,
0.000092, 3.73, 0.000096, 3.72, 0.000100, 3.71, 0.000104, 3.70, 0.000108, 3.69, 0.000112, 3.68, 0.000117, 3.67, 0.000121, 3.66, 0.000126,
3.65, 0.000131, 3.64, 0.000136, 3.63, 0.000142, 3.62, 0.000147, 3.61, 0.000153, 3.60, 0.000159, 3.59, 0.000165, 3.58, 0.000172, 3.57,
0.000178, 3.56, 0.000185, 3.55, 0.000193, 3.54, 0.000200, 3.53, 0.000208, 3.52, 0.000216, 3.51, 0.000224, 3.50, 0.000233, 3.49, 0.000242,
3.48, 0.000251, 3.47, 0.000260, 3.46, 0.000270, 3.45, 0.000280, 3.44, 0.000291, 3.43, 0.000302, 3.42, 0.000313, 3.41, 0.000325, 3.40,
0.000337, 3.39, 0.000349, 3.38, 0.000362, 3.37, 0.000376, 3.36, 0.000390, 3.35, 0.000404, 3.34, 0.000419, 3.33, 0.000434, 3.32, 0.000450,
3.31, 0.000466, 3.30, 0.000483, 3.29, 0.000501, 3.28, 0.000519, 3.27, 0.000538, 3.26, 0.000557, 3.25, 0.000577, 3.24, 0.000598, 3.23,
0.000619, 3.22, 0.000641, 3.21, 0.000664, 3.20, 0.000687, 3.19, 0.000711, 3.18, 0.000736, 3.17, 0.000762, 3.16, 0.000789, 3.15, 0.000816,
3.14, 0.000845, 3.13, 0.000874, 3.12, 0.000904, 3.11, 0.000935, 3.10, 0.000968, 3.09, 0.001001, 3.08, 0.001035, 3.07, 0.001070, 3.06,
0.001107, 3.05, 0.001144, 3.04, 0.001183, 3.03, 0.001223, 3.02, 0.001264, 3.01, 0.001306, 3.00, 0.001350, 2.99, 0.001395, 2.98, 0.001441,
2.97, 0.001489, 2.96, 0.001538, 2.95, 0.001589, 2.94, 0.001641, 2.93, 0.001695, 2.92, 0.001750, 2.91, 0.001807, 2.90, 0.001866, 2.89,
0.001926, 2.88, 0.001988, 2.87, 0.002052, 2.86, 0.002118, 2.85, 0.002186, 2.84, 0.002256, 2.83, 0.002327, 2.82, 0.002401, 2.81, 0.002477,
2.80, 0.002555, 2.79, 0.002635, 2.78, 0.002718, 2.77, 0.002803, 2.76, 0.002890, 2.75, 0.002980, 2.74, 0.003072, 2.73, 0.003167, 2.72,
0.003264, 2.71, 0.003364, 2.70, 0.003467, 2.69, 0.003573, 2.68, 0.003681, 2.67, 0.003793, 2.66, 0.003907, 2.65, 0.004025, 2.64, 0.004145,
2.63, 0.004269, 2.62, 0.004396, 2.61, 0.004527, 2.60, 0.004661, 2.59, 0.004799, 2.58, 0.004940, 2.57, 0.005085, 2.56, 0.005234, 2.55,
```

```

0.005386, 2.54, 0.005543, 2.53, 0.005703, 2.52, 0.005868, 2.51, 0.006037, 2.50, 0.006210, 2.49, 0.006387, 2.48, 0.006569, 2.47, 0.006756,
2.46, 0.006947, 2.45, 0.007143, 2.44, 0.007344, 2.43, 0.007549, 2.42, 0.007760, 2.41, 0.007976, 2.40, 0.008198, 2.39, 0.008424, 2.38,
0.008656, 2.37, 0.008894, 2.36, 0.009137, 2.35, 0.009387, 2.34, 0.009642, 2.33, 0.009903, 2.32, 0.010170, 2.31, 0.010444, 2.30, 0.010724,
2.29, 0.011011, 2.28, 0.011304, 2.27, 0.011604, 2.26, 0.011911, 2.25, 0.012224, 2.24, 0.012545, 2.23, 0.012874, 2.22, 0.013209, 2.21,
0.013553, 2.20, 0.013903, 2.19, 0.014262, 2.18, 0.014629, 2.17, 0.015003, 2.16, 0.015386, 2.15, 0.015778, 2.14, 0.016177, 2.13, 0.016586,
2.12, 0.017003, 2.11, 0.017429, 2.10, 0.017864, 2.09, 0.018309, 2.08, 0.018763, 2.07, 0.019226, 2.06, 0.019699, 2.05, 0.020182, 2.04,
0.020675, 2.03, 0.021178, 2.02, 0.021692, 2.01, 0.022216, 2.00, 0.022750, 1.99, 0.023295, 1.98, 0.023852, 1.97, 0.024419, 1.96, 0.024998,
1.95, 0.025588, 1.94, 0.026190, 1.93, 0.026803, 1.92, 0.027429, 1.91, 0.028067, 1.90, 0.028717, 1.89, 0.029379, 1.88, 0.030054, 1.87,
0.030742, 1.86, 0.031443, 1.85, 0.032157, 1.84, 0.032884, 1.83, 0.033625, 1.82, 0.034380, 1.81, 0.035148, 1.80, 0.035930, 1.79, 0.036727,
1.78, 0.037538, 1.77, 0.038364, 1.76, 0.039204, 1.75, 0.040059, 1.74, 0.040930, 1.73, 0.041815, 1.72, 0.042716, 1.71, 0.043633, 1.70,
0.044565, 1.69, 0.045514, 1.68, 0.046479, 1.67, 0.047460, 1.66, 0.048457, 1.65, 0.049471, 1.64, 0.050503, 1.63, 0.051551, 1.62, 0.052616,
1.61, 0.053699, 1.60, 0.054799, 1.59, 0.055917, 1.58, 0.057053, 1.57, 0.058208, 1.56, 0.059380, 1.55, 0.060571, 1.54, 0.061780, 1.53,
0.063008, 1.52, 0.064255, 1.51, 0.065522, 1.50, 0.066807, 1.49, 0.068112, 1.48, 0.069437, 1.47, 0.070781, 1.46, 0.072145, 1.45, 0.073529,
1.44, 0.074934, 1.43, 0.076359, 1.42, 0.077804, 1.41, 0.079270, 1.40, 0.080757, 1.39, 0.082264, 1.38, 0.083793, 1.37, 0.085343, 1.36,
0.086915, 1.35, 0.088508, 1.34, 0.090123, 1.33, 0.091759, 1.32, 0.093418, 1.31, 0.095098, 1.30, 0.096800, 1.29, 0.098525, 1.28, 0.100273,
1.27, 0.102042, 1.26, 0.103835, 1.25, 0.105650, 1.24, 0.107488, 1.23, 0.109349, 1.22, 0.111232, 1.21, 0.113139, 1.20, 0.115070, 1.19,
0.117023, 1.18, 0.119000, 1.17, 0.121000, 1.16, 0.123024, 1.15, 0.125072, 1.14, 0.127143, 1.13, 0.129238, 1.12, 0.131357, 1.11, 0.133500,
1.10, 0.135666, 1.09, 0.137857, 1.08, 0.140071, 1.07, 0.142310, 1.06, 0.144572, 1.05, 0.146859, 1.04, 0.149170, 1.03, 0.151505, 1.02,
0.153864, 1.01, 0.156248, 1.00, 0.158655, 0.99, 0.161087, 0.98, 0.163543, 0.97, 0.166023, 0.96, 0.168528, 0.95, 0.171056, 0.94, 0.173609,
0.93, 0.176186, 0.92, 0.178786, 0.91, 0.181411, 0.90, 0.184060, 0.89, 0.186733, 0.88, 0.189430, 0.87, 0.192150, 0.86, 0.194895, 0.85,
0.197663, 0.84, 0.200454, 0.83, 0.203269, 0.82, 0.206108, 0.81, 0.208970, 0.80, 0.211855, 0.79, 0.214764, 0.78, 0.217695, 0.77, 0.220650,
0.76, 0.223627, 0.75, 0.226627, 0.74, 0.229650, 0.73, 0.232695, 0.72, 0.235762, 0.71, 0.238852, 0.70, 0.241964, 0.69, 0.245097, 0.68,
0.248252, 0.67, 0.251429, 0.66, 0.254627, 0.65, 0.257846, 0.64, 0.261086, 0.63, 0.264347, 0.62, 0.267629, 0.61, 0.270931, 0.60, 0.274253,
0.59, 0.277595, 0.58, 0.280957, 0.57, 0.284339, 0.56, 0.287740, 0.55, 0.291160, 0.54, 0.294599, 0.53, 0.298056, 0.52, 0.301532, 0.51,
0.305026, 0.50, 0.308538, 0.49, 0.312067, 0.48, 0.315614, 0.47, 0.319178, 0.46, 0.322758, 0.45, 0.326355, 0.44, 0.329969, 0.43, 0.333598,
0.42, 0.337243, 0.41, 0.340903, 0.40, 0.344578, 0.39, 0.348268, 0.38, 0.351973, 0.37, 0.355691, 0.36, 0.359424, 0.35, 0.363169, 0.34,
0.366928, 0.33, 0.370700, 0.32, 0.374484, 0.31, 0.378280, 0.30, 0.382089, 0.29, 0.385908, 0.28, 0.389739, 0.27, 0.393580, 0.26, 0.397432,
0.25, 0.401294, 0.24, 0.405165, 0.23, 0.409046, 0.22, 0.412936, 0.21, 0.416834, 0.20, 0.420740, 0.19, 0.424655, 0.18, 0.428576, 0.17,
0.432505, 0.16, 0.436441, 0.15, 0.440382, 0.14, 0.444330, 0.13, 0.448283, 0.12, 0.452242, 0.11, 0.456205, 0.10, 0.460172, 0.09, 0.464144,
0.08, 0.468119, 0.07, 0.472097, 0.06, 0.476078, 0.05, 0.480061, 0.04, 0.484047, 0.03, 0.488034, 0.02, 0.492022, 0.01, 0.496011, 0.00,
0.500000, 0.00)), 6))) end P_value_two_tailed_test, non_zero_plots_x, non_zero_plots_y, variance_x, variance_y, covariance_xy
covariance_xy_final, cov_correction, total_plots, total_population_acres from (select estimate_y, non_zero_plots_y, estimate_x,
non_zero_plots_x, difference_x_y, variance_x, variance_y, (covariance_xy * cov_correction) covariance_xy, cov_correction, case when
(((variance_x) + (variance_y) - (2 * (covariance_xy * cov_correction))) = 0) then 0 else (sqrt(round(((variance_x) + (variance_y) - (2 *
(covariance_xy * cov_correction)))), 2))) end std_error_difference, total_plots, total_population_acres, group_evalgrps from (select
sum(estimate_Y_est_unit) estimate_y, sum(non_zero_plots_y_est_unit) non_zero_plots_y, sum(estimate_x_est_unit) estimate_x,
sum(non_zero_plots_x_est_unit) non_zero_plots_x, round(sum(nvl(estimate_x_est_unit, 0)) - sum(nvl(estimate_y_est_unit, 0)), 3)
Difference_x_y, sum(variance_x_est_unit) variance_x, sum(variance_y_est_unit) variance_y, sum(covariance_xy_est_unit) covariance_xy,
(select case when domain_x_cnt = 0 or domain_y_cnt = 0 then 0 else ((domain_xy_cnt / domain_x_cnt) * (domain_xy_cnt / domain_y_cnt))
end cov_correction from (select sum(case when domain_x > 0 then 1 else 0 end) domain_x_cnt, sum(case when domain_y > 0 then 1 else 0
end) domain_y_cnt, sum(case when domain_x > 0 and domain_y > 0 then 1 else 0 end) domain_xy_cnt from (select plt_cn, sum(domain_x)
domain_x, sum(domain_y) domain_y from (select p.cn plt_cn, decode(domain, 'X', 1, 0) domain_x, decode(domain, 'Y', 1, 0) domain_y FROM
FS_FIADB.plot p, FS_FIADB.pop_plot_stratum_assgn pop_plot_stratum_assgn, FS_FIADB.pop_stratum pop_stratum, FS_FIADB.pop_estn_unit
pop_estn_unit, FS_FIADB.pop_eval pop_eval, FS_FIADB.pop_eval_grp pop_eval_grp, FS_FIADB.pop_eval_attribute pop_eval_attribute,
FS_FIADB.pop_eval_typ pet, (select distinct 'Y' domain, pop_estn_unit.cn pop_estn_unit_cn, pop_eval.cn pop_eval_cn from FS_FIADB.plot p,
FS_FIADB.cond c, FS_FIADB.pop_plot_stratum_assgn pop_plot_stratum_assgn, FS_FIADB.pop_stratum pop_stratum,
FS_FIADB.pop_estn_unit pop_estn_unit, FS_FIADB.pop_eval pop_eval, FS_FIADB.pop_eval_grp pop_eval_grp, FS_FIADB.pop_eval_attribute
pop_eval_attribute, FS_FIADB.pop_eval_typ pet, FS_FIADB.subp_cond_chng_mtrx sccm, FS_FIADB.cond c1 where p.cn = c.plt_cn and
pop_plot_stratum_assgn.plt_cn = p.cn and pop_plot_stratum_assgn.stratum_cn = pop_stratum.cn and pop_estn_unit.cn =
pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn = pop_eval_attribute.eval_cn and pop_eval.cn =
pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval.grp.eval_grp in (102015) and pop_eval_attribute.attribute_nbr = 126 and
pet.eval_typ = 'EXPCHNG' and sccm.plt_cn = c.plt_cn and sccm.condid = c.condid and sccm.prev_plt_cn = c1.plt_cn and sccm.prevcond =
c1.condid and decode(c.prop_basis, 'MACR', 3, 1) = sccm.subptyp and nvl(c.cond_nonsample_reasn_cd, 0) = 0 and
nvl(c1.cond_nonsample_reasn_cd, 0) = 0 and c1.cond_status_cd = 1 union select distinct 'X' domain, pop_estn_unit.cn pop_estn_unit_cn,
pop_eval.cn pop_eval_cn from FS_FIADB.plot p, FS_FIADB.cond c, FS_FIADB.pop_plot_stratum_assgn pop_plot_stratum_assgn,
FS_FIADB.pop_stratum pop_stratum, FS_FIADB.pop_estn_unit pop_estn_unit, FS_FIADB.pop_eval pop_eval, FS_FIADB.pop_eval_grp
pop_eval_grp, FS_FIADB.pop_eval_attribute pop_eval_attribute, FS_FIADB.pop_eval_typ pet, FS_FIADB.subp_cond_chng_mtrx sccm,
```

```

FS_FIADB.cond c1 where p.cn = c.plt_cn and pop_plot_stratum_assgn.plt_cn = p.cn and pop_plot_stratum_assgn.stratum_cn =
pop_stratum.cn and pop_estn_unit.cn = pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn =
pop_eval_attribute.eval_cn and pop_eval.cn = pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval_grp.eval_grp in (102015)
and pop_eval_attribute.attribute_nbr = 126 and pet.eval_typ = 'EXPCHNG' and sccm.plt_cn = c.plt_cn and sccm.condid = c.condid and
sccm.prev_plt_cn = c1.plt_cn and sccm.prevcond = c1.condid and decode(c.prop_basis, 'MACR', 3, 1) = sccm.subptyp and
nvl(c.cond_nonsample_reasn_cd, 0) = 0 and nvl(c1.cond_nonsample_reasn_cd, 0) = 0 and c.cond_status_cd = 1) domains_est_units where
pop_plot_stratum_assgn.plt_cn = p.cn and pop_plot_stratum_assgn.stratum_cn = pop_stratum.cn and pop_estn_unit.cn =
pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn = pop_eval_attribute.eval_cn and pop_eval.cn =
pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval.cn = pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and
pop_eval_attribute.attribute_nbr = 126 and pet.eval_typ = 'EXPCHNG' and pop_eval_grp.eval_grp in (102015) and
domains_est_units.pop_estn_unit_cn = pop_estn_unit.cn and domains_est_units.pop_eval_cn = pop_eval.cn group by
domains_est_units.domain, p.cn order by domain_x, domain_y group by plt_cn) sum_domain_combo_counts) cov_correction,
sum(total_plots_est_unit) total_plots, sum(total_area_est_unit) total_population_acres, Group_evalgrps from (select sum(nvl(ysum_hd, 0) *
expns) estimate_Y_est_unit, sum(Non_zero_plots_y) NON_ZERO_PLOTS_y_est_unit, sum(nvl(xsum_hd, 0) * expns) estimate_x_est_unit,
sum(Non_zero_plots_x) NON_ZERO_PLOTS_x_est_unit, pop_eval_grp_cn, eval_grp, eval_grp_descr, estn_unit_cn, sum(n_h)
total_plots_est_unit, total_area_est_unit, (total_area_est_unit * total_area_est_unit / SUM(n_h) * ((SUM(w_h * n_h * (((nvl(ysum_hd_sqr, 0) /
n_h) - (nvl(ysum_hd, 0) / n_h) * (nvl(ysum_hd, 0) / n_h)) / (n_h - 1))) + 1 / SUM(n_h) * (SUM((1 - w_h) * n_h * (((nvl(ysum_hd_sqr, 0) /
n_h) - (nvl(ysum_hd, 0) / n_h) * (nvl(ysum_hd, 0) / n_h)) / (n_h - 1)))) variance_y_est_unit, (total_area_est_unit * total_area_est_unit /
SUM(n_h) * ((SUM(w_h * n_h * (((nvl(xsum_hd_sqr, 0) / n_h) - (nvl(xsum_hd, 0) / n_h) * (nvl(xsum_hd, 0) / n_h)) / (n_h - 1))) + 1 /
SUM(n_h) * (SUM((1 - w_h) * n_h * (((nvl(xsum_hd_sqr, 0) / n_h) - (nvl(xsum_hd, 0) / n_h) * (nvl(xsum_hd, 0) / n_h)) / (n_h - 1)))) variance_x_est_unit, (total_area_est_unit * total_area_est_unit / SUM(n_h) * ((SUM(w_h * n_h * (((nvl(ysum_hd, 0) / n_h) -
(nvl(ysum_hd, 0) / n_h)) / (n_h - 1))) + 1 / SUM(n_h) * (SUM((1 - w_h) * n_h * (((nvl(ysum_hd, 0) / n_h) - (nvl(ysum_hd, 0) / n_h) *
(nvl(ysum_hd, 0) / n_h)) / (n_h - 1)))) covariance_xy_est_unit, 1 group_evalgrps from (select eval_grp_descr, eval_grp,
pop_eval_grp_cn, estn_unit_cn, pop_stratum_Cn cn, sum(expns) expns, sum(w_h) w_h, sum(total_area_est_unit) total_area_est_unit,
sum(n_h) n_h, sum(ysum_hd) ysum_hd, sum(ysum_hd_sqr) ysum_hd_sqr, sum(xsum_hd) xsum_hd, sum(xsum_hd_sqr) xsum_hd_sqr,
sum(xysum_hd) xysum_hd, sum(plot_cnt_eval_typ) plot_cnt_eval_typ, sum(non_zero_plots_y) non_zero_plots_y, sum(non_zero_plots_x)
non_zero_plots_x from ( select pop_eval_grp.eval_grp_descr, pop_eval_grp.eval_grp, pop_eval_grp.pop_eval_grp_cn,
pop_stratum.estn_unit_cn, pop_stratum.cn pop_stratum_cn, pop_stratum.expns expns, p1pointcnt / (select sum(strs.p1pointcnt) from
FS_FIADB.POP_stratum strs where strs.estn_unit_cn = pop_stratum.estn_unit_cn) w_h, (select sum(eu_s.area_used) from
FS_FIADB.POP_ESTN_UNIT eu_s where eu_s.cn = pop_stratum.estn_unit_cn) total_area_est_unit, pop_stratum.p2pointcnt n_h, 0 ysum_hd, 0
ysum_hd_sqr, 0 xsum_hd, 0 xsum_hd_sqr, 0 xysum_hd, 0 plot_cnt_eval_typ, 0 non_zero_plots_y, 0 non_zero_plots_x FROM
FS_FIADB.pop_stratum pop_stratum, FS_FIADB.pop_estn_unit pop_estn_unit, FS_FIADB.pop_eval pop_eval, FS_FIADB.pop_eval_grp
pop_eval_grp, FS_FIADB.pop_eval_typ pet, FS_FIADB.pop_eval_attribute pop_eval_attribute where pop_estn_unit.cn =
pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn = pet.eval_cn and pop_eval_attribute.eval_cn =
pop_eval.cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval.cn = pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and
pop_eval_grp.eval_grp in (102015) and pop_eval_attribute.attribute_nbr = 126 and pet.eval_typ = 'EXPCHNG' union select
all_plots.eval_grp_descr, all_plots.eval_grp, all_plots.eval_grp.pop_eval_grp_cn, all_plots.estn_unit_cn, all_plots.pop_stratum_cn, sum(0)
expns, sum(0) w_h, sum(0) total_area_est_unit, sum(0) n_h, sum(nvl(y_hid_adjusted, 0)) ysum_hd, sum(nvl(y_hid_adjusted *
y_hid_adjusted, 0)) ysum_hd_sqr, sum(nvl(x_hid_adjusted, 0)) xsum_hd, sum(nvl(x_hid_adjusted * x_hid_adjusted, 0)) xsum_hd_sqr,
sum(nvl(x_hid_adjusted, 0) * nvl(y_hid_adjusted, 0)) xysum_hd, sum(all_plots.plot_count) plot_cnt_eval_typ, sum(decode(y_hid_adjusted, 0,
0, null, 0, 1)) non_zero_plots_y, sum(decode(x_hid_adjusted, 0, 0, null, 0, 1)) non_zero_plots_x from (select pop_eval_grp.eval_grp_descr,
pop_eval_grp.pop_eval, pop_eval.grp_cn, pop_stratum.estn_unit_cn, pop_stratum.cn pop_stratum_cn, p.cn plt_cn, pop_eval.cn
pop_eval_cn, 1 plot_count from FS_FIADB.plot p, FS_FIADB.cond c, FS_FIADB.pop_plot_stratum_assgn pop_plot_stratum_assgn,
FS_FIADB.pop_stratum pop_stratum, FS_FIADB.pop_estn_unit pop_estn_unit, FS_FIADB.pop_eval pop_eval, FS_FIADB.pop_eval_grp
pop_eval_grp, FS_FIADB.pop_eval_attribute pop_eval_attribute, FS_FIADB.pop_eval_typ pet where p.cn = c.plt_cn and
pop_plot_stratum_assgn.plt_cn = p.cn and pop_plot_stratum_assgn.stratum_cn = pop_stratum.cn and pop_estn_unit.cn =
pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn = pop_eval_attribute.eval_cn and
pop_eval.cn = pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval.grp.cn = pop_eval.eval_grp in (102015) and
FS_FIADB.pop_eval_attribute.pop_eval_attribute, FS_FIADB.pop_eval_typ pet, FS_FIADB.subp_cond_chng_mtrc sccm, FS_FIADB.cond c1
where p.cn = c.plt_cn and pop_plot_stratum_assgn.plt_cn = p.cn and pop_plot_stratum_assgn.stratum_cn = pop_stratum.cn and
pop_estn_unit.cn = pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn = pop_eval_attribute.eval_cn and
pop_eval.cn = pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval.grp.cn = pop_eval.eval_grp in (102015) and

```

```

pop_eval_attribute.attribute_nbr = 126 and pet.eval_typ = 'EXPCHNG' and sccm.plt_cn = c.plt_cn and sccm.condid = c.condid and
sccm.prev_plt_cn = c1.plt_cn and sccm.prevcond = c1.condid and decode(c.prop_basis, 'MACR', 3, 1) = sccm.subptyp and
nvl(c.cond_nonsample_reasn_cd, 0) = 0 and nvl(c1.cond_nonsample_reasn_cd, 0) = 0 and c1.cond_status_cd = 1 group by pop_stratum.cn,
pop_eval.cn, p.cn) y_plot,(select pop_stratum.cn pop_stratum_cn, p.cn plt_cn, pop_eval.cn pop_eval_cn,
SUM(nvl(sccm.SUBPTYP_PROP_CHNG / 4 * decode(c.prop_basis, 'MACR', pop_stratum.adj_factor_macr, pop_stratum.adj_factor_subp), 0))
x_hid_adjusted from FS_FIADB.plot p, FS_FIADB.cond c, FS_FIADB.pop_plot_stratum_assgn pop_plot_stratum_assgn, FS_FIADB.pop_stratum
pop_stratum, FS_FIADB.pop_estn_unit pop_estn_unit, FS_FIADB.pop_eval pop_eval, FS_FIADB.pop_eval_grp pop_eval_grp,
FS_FIADB.pop_eval_attribute pop_eval_attribute, FS_FIADB.pop_eval_typ pet, FS_FIADB.subp_cond_chng_mtx sccm, FS_FIADB.cond c1
where p.cn = c.plt_cn and pop_plot_stratum_assgn.plt_cn = p.cn and pop_plot_stratum_assgn.stratum_cn = pop_stratum.cn and
pop_estn_unit.cn = pop_stratum.estn_unit_cn and pop_eval.cn = pop_estn_unit.eval_cn and pop_eval.cn = pop_eval_attribute.eval_cn and
pop_eval.cn = pet.eval_cn and pet.eval_grp_cn = pop_eval_grp.cn and pop_eval_grp.eval_grp in (102015) and
pop_eval_attribute.attribute_nbr = 126 and pet.eval_typ = 'EXPCHNG' and sccm.plt_cn = c.plt_cn and sccm.condid = c.condid and
sccm.prev_plt_cn = c1.plt_cn and sccm.prevcond = c1.condid and decode(c.prop_basis, 'MACR', 3, 1) = sccm.subptyp and
nvl(c.cond_nonsample_reasn_cd, 0) = 0 and nvl(c1.cond_nonsample_reasn_cd, 0) = 0 and c1.cond_status_cd = 1 group by pop_stratum.cn,
pop_eval.cn, p.cn) x_plot where all_plots.plt_cn = x_plot.plt_cn(+) and all_plots.plt_cn = y_plot.plt_cn(+) and all_plots.pop_eval_cn =
y_plot.pop_eval_cn(+) and all_plots.pop_eval_cn = x_plot.pop_eval_cn(+) group by all_plots.eval_grp_descr, all_plots.eval_grp,
all_plots.eval_grp_cn, all_plots.estn_unit_cn, all_plots.pop_stratum_cn ) group by eval_grp_descr, eval_grp, pop_eval_grp_cn, estn_unit_cn,
pop_stratum_cn ) estimate_by_estn_unit_strata group by pop_eval_grp_cn, eval_grp, eval_grp_descr, estn_unit_cn, total_area_est_unit
by_est_unit group by Group_evalgrps total_estimates) std_error_difference

```

RESTful Web Service Call

This RESTful Web Service call can be used to generate estimates directly from browser address line

http://localhost:8080/Validator/rest/Validator/diftest?estType=0079_Y&evalGrp=102015&constraintT2=and%20c.cond_status_cd%20=%201&constraintT1=and%20c1.cond_status_cd%20=%201&schemaName=FS_FIADB.&outputFormat=HTML